ĐẠI HỌC BÁCH KHOA HÀ NỘI

**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

----- 🙡 🕮 🙣 -----



**BÁO CÁO PROJECT I**

***Báo cáo tuần 2***

Giảng viên: **Ngô Lam Trung**

Sinh viên thực hiện: Nguyễn Quang Long

Mã số sinh viên: 20194320

**Hà Nội, năm 2021**

Nội dung đã hoàn thành:

**38. Concatenate Arrays**

a. Đề bài:

Cho 2 mảng số nguyên a và b, nối mảng b vào sau mảng a.

b. Giải pháp:

Sử dụng hàm có sẵn insert().

c. Code:

vector<int> concatenateArrays(vector<int> a, vector<int> b) {

    a.insert(end(a), begin(b), end(b));

    return a;

}

**39. Remove Array Part**

a. Đề bài:

Xóa 1 phần của mảng đã cho từ vị trí l đến r

b. Giải pháp:

Sử dụng hàm splice()

c. Code:

function removeArrayPart(inputArray, l, r) {

  inputArray.splice(l, r - l + 1);

  return inputArray;

}

**40. Is Smooth?**

a. Đề bài:

Cho 1 mảng, middle của mảng được xác định bằng cách:

-Nếu mảng chứa 1 số lẻ phần tử thì middle là phần tử có chỉ số giống nhau khi đếm từ đầu mảng hoặc cuối mảng

-Nếu mảng chứa 1 số chẵn phần từ thì middle là tổng của 2 phần tử có chỉ số hơn kém nhau 1 khi đếm từ đầu mảng và cuối mảng

Mảng được gọi là Smooth nếu phần tử đầu bằng phần tử cuối và bằng middle

Xác định xem mảng đã cho có Smooth không?

b. Giải pháp:

-Tìm phần tử middle của mảng

-Nếu phần tử đầu và phần tử cuối của mảng đều bằng middle thì mảng Smooth, ngược lại thì không.

c. Code:

function isSmooth(arr) {

    var mid;

    if(arr.length%2==0){

        mid = arr[arr.length/2] + arr[(arr.length/2)-1]

    } else {

        mid = arr[Math.floor(arr.length/2)]

    }

    return arr[0] == mid && mid == arr[arr.length-1]

}

**41. Replace Middle**

a. Đề bài:

Cho 1 mảng, middle của mảng được xác định bằng cách:

-Nếu mảng chứa 1 số lẻ phần tử thì middle là phần tử có chỉ số giống nhau khi đếm từ đầu mảng hoặc cuối mảng

-Nếu mảng chứa 1 số chẵn phần từ thì middle là tổng của 2 phần tử có chỉ số hơn kém nhau 1 khi đếm từ đầu mảng và cuối mảng

Nếu middle là tổng của 2 phần tử, thay thế 2 phần tử này bằng middle

b. Giải pháp:

Nếu mảng chứa 1 số chẵn phần tử:

-Tìm vị trí của 2 phần tử đứng giữa

-Tìm giá trị middle

- Thay thế 2 phần tử đứng giữa bằng middle

Còn nếu mảng chứa 1 số lẻ phần tử thì không phải làm gì

c. Code:

function replaceMiddle(arr) {

  if (arr.length % 2 === 0) {

    var midPos = arr.length / 2;

    var mid = arr[midPos] + arr[midPos - 1];

    arr.splice(midPos - 1, 2, mid);

  }

  return arr;

}

**42. Make Array Consecutive 2**

a. Đề bài:

Cho 1 mảng các số nguyên không âm

Cần thêm vào mảng bao nhiêu số nguyên để được 1 số nguyên liên tiếp từ phần tử bé nhất đến phần tử lớn nhất của mảng?

b. Giải pháp:

-Tính số lượng số nguyên từ phần tử bé nhất đến lớn nhất của mảng rồi trừ đi số lượng phần tử của mảng

c. Code:

function makeArrayConsecutive2(statues) {

    return Math.max(...statues)-Math.min(...statues)+1-statues.length;

}

**43. Is power?**

a. Đề bài:

Xác định xem số đã cho có phải là lũy thừa của một số nguyên không âm không?

b. Giải pháp:

Tính lũy thừa của các số từ 2 đến căn n rồi so sánh với n

c. Code:

bool isPower(int n)

{

    if (n == 1) return true;

    for(int t = 2; t <= sqrt(n); ++t){

        int k=t;

        while (k<n) {

            k=k\*t;

        }

        if(k==n) return true;

    }

    return false;

}

**44. Is sum of consecutive 2?**

a. Đề bài:

Tính số cách biểu diễn 1 số nguyên bằng tổng các số nguyên dương liên tiếp

b. Giải pháp:

-Tính tổng các dãy từ 1,2,.. đến n/2 rồi so sánh với n

c. Code

int isSumOfConsecutive2(int n) {

    int count =0;

    for(int i=1;i<=n/2;i++){

        int s=0;

        for(int j=i;j<=n/2+1;j++){

            s+=j;

            if(s==n) {count++;

                        break;}

        }

    }

    return count;

}

**45. Square Digits Sequence**

a. Đề bài:

Cho 1 số nguyên dương, phần tử đằng sau được xác định bằng tổng bình phương các chữ số của phần tử đứng trước, dãy dừng lại khi xuất hiện phần tử giống với phần tử đã có trong dãy. Tính số phần tử của dãy.

b. Giải pháp

Tìm phần tử tiếp theo của dãy, nếu phần tử này đã tồn tại thì dãy kết thúc còn không thì thêm phần tử vào dãy lại tìm phần tử tiếp theo. Kết quả là số phần tử của dãy +1.

c. Code

int check(int n){

    if(n<10) return pow(n,2);

    else return pow(n%10,2)+check(n/10);

}

int squareDigitsSequence(int a0) {

    int n=a0;

    set<int> sequence={n};

    while(1){

        n=check(n);

        if(sequence.count(n)) return size(sequence)+1;

        else sequence.insert(n);

    }

}

**46. Pages Numbering With Ink**

a. Đề bài

Tính số trang cuối cùng có thể đánh số cho sách biết trang hiện tại cần đánh là current và có thể đánh numberOfDigits chữ số nữa.

b. Giải pháp

Nếu số chữ số có thể đánh nhỏ hơn số chữ số của current thì trả về số trang đứng trước current, còn không gọi đệ quy với trang tiếp theo của current và số chữ số trừ đi số chữ số của current.

c. Code

int numberDegree(int n){

    if(n<10) return 1;

    else return 1+numberDegree(n/10);

}

int pagesNumberingWithInk(int current, int numberOfDigits) {

    if(numberOfDigits<numberDegree(current)) return current-1;

    else return pagesNumberingWithInk(current+1, numberOfDigits-numberDegree(current));

}

**47. Comfortable Numbers**

a. Đề bài

Số a được gọi là comfortable với số b nếu a khác b và b nằm trên đoạn [a - s(a), a + s(a)] với s(x) là tổng các chữ số của x.

Có bao nhiêu cặp (a,b) với a<b cùng nằm trên đoạn [l,r] và mỗi số comfortable với số còn lại?

b. Giải pháp

Duyệt tất cả các cặp (a,b) từ l đến r, nếu thỏa mãn thì chọn

c. Code

int s(int x){

    if(x<10) return x;

    else return x%10+s(x/10);

}

int comfortableNumbers(int L, int R) {

    int pairs = 0;

    for(int i=L; i<R; i++){

        for(int j=i+1; j<=R; j++){

            if (i<=(j+s(j)) && i>=(j-s(j)) && j<=(i+s(i)) && j>=(i-s(i))) pairs++;

            }

        }

     return pairs;

}

**48. Weak Number**

a. Đề bài

Điểm yếu của số x được xác định bằng số lượng các số nguyên dương nhỏ hơn x có nhiều ước hơn x

Số này được gọi là yếu hơn số kia nếu có nhiều điểm yếu hơn.

Cho một số nguyên n, có bao nhiêu điểm yếu của số yếu nhất trong đoạn [1,n] và có bao nhiêu số trong đoạn này có cùng số điểm yếu này?

b. Giải pháp

Duyệt toàn bộ các số trong đoạn [1,n], tìm số yếu nhất, nếu có số có cùng điểm yếu thì tăng biến đếm thêm 1

c. Code

int numberofDivisors(int n){

    int count=0;

    for(int i=1;i<=n;i++){

        if(n%i==0) count++;

    }

    return count;

}

int weakness(int x){

    int x\_div= numberofDivisors(x),count=0;

    for(int i=1;i<x;i++){

        if(numberofDivisors(i)>x\_div) count++;

    }

    return count;

}

vector<int> weakNumbers(int n) {

    vector<int> result;

    int weakness\_max=0;

    int count=0;

    for(int i=1;i<=n;i++){

        if(weakness(i)>=weakness\_max){

            if (weakness(i)==weakness\_max) count++;

            else {weakness\_max=weakness(i);

                    count=1;}

        }

    }

    result.push\_back(weakness\_max);

    result.push\_back(count);

    return result;

}

49. Rectangle Rotation

a. Đề bài:

Một hình chữ nhật có các cạnh là số nguyên chẵn được vẽ trên mặt phẳng Đề-các. Tâm hình chữ nhật trùng với điểm (0,0) nhưng các cạnh tạo với các trục 1 góc 45 độ

Có bao nhiêu điểm có tọa độ nguyên nằm trong hình chữ nhật (kể cả các cạnh của nó)?

b. Giải pháp:

Câu này em xem lời giải trên mạng nhưng chưa hiểu ạ

c. Code

int rectangleRotation(int a, int b) {

    a/=1.414;

    b/=1.414;

    return 2\*a\*b+a+b|1;

}

**50. Crossword Formation**

a. Đề bài

Cho 4 từ, tìm số cách để tạo ra ô chữ theo các yêu cầu:

-Ô chữ phải chứa đủ 4 từ

- 4 từ này tạo nên giao điểm theo cặp

- Tất cả các từ phải viết từ trái qua phải hoặc từ trên xuống dưới

- Diện tích hình chữ nhật được tạo bởi các ô trống bên trong các giao điểm không được bằng 0

b. Giải pháp

**51. Enclose in Brackets**

a. Đề bài

Cho 1 xâu, thêm đóng mở ngoặc ( ) vào xâu đó

b. Giải pháp

Thêm ( vào đầu xâu và ) vào cuối xâu

c. Code

string encloseInBrackets(string inputString) {

    return "("+inputString+")";

}

**52. Proper Noun Correction**

a. Đề bài

Cho 1 xâu, chuyển chữ cái đầu tiên thành viết hoa và các chữ cái còn lại viết thường

b. Giải pháp

Chuyển tất cả các chữ cái thành viết thường rồi chuyển chữ cái đầu tiên thành viết hoa

c. Code

string properNounCorrection(string noun) {

    for(int i=1;i<noun.length();i++){

       noun[i]= tolower(noun[i]);

    }

     noun[0]=toupper(noun[0]);

    return noun;

}

**53. Is Tandem Repeat?**

a. Đề bài

Xác định xem xâu đã cho có phải do 1 xâu ghép với chính nó tạo thành không?

b. Giải pháp

Nếu độ dài xâu là lẻ thì không thỏa mãn

Ngược lại, tách xâu đã cho thành 2 xâu con độ dài bằng nhau và bằng 1 nửa xâu đã cho, nếu 2 xâu con là giống nhau thì thỏa mãn đề bài.

c. Code

bool isTandemRepeat(string inputString) {

    if(inputString.length()%2==1) return false;

    else{

 string sub1= inputString.substr(0,inputString.length()/2);

 string sub2=inputString.substr(inputString.length()/2,inputString.length()/2);

 if(sub1==sub2) return true;

 else return false;

}

}

**54. Is Case-Insensitive Palindrome?**

a. Đề bài

Cho 1 chuỗi, kiểm tra xem nó có thể trở thành chuỗi Palindrome nếu thay đổi kiểu chữ của 1 vài (hoặc không) ký tự?

b. Giải pháp

Duyệt lần lượt các cặp ký tự tương ứng xem có phải các ký tự giống nhau không, nếu có cặp ký tự khác nhau thì trả về false.

c. Code

#include <cwctype>

#include <ios>

bool isCaseInsensitivePalindrome(string inputString) {

    int begin =0;

    int end = inputString.length()-1;

    while(begin<end){

        if(tolower(inputString[begin])!=tolower(inputString[end])) return false;

        begin ++;

        end--;

    }

    return true;

}

**55. Find Email Domain**

a. Đề bài

Cho 1 địa chỉ email hợp lệ, tìm tên miền của địa chỉ email đó.

b. Giải pháp

Tìm đến ký tự @ đầu tiên mà sau đó là chữ cái, tên miền sẽ là xâu con tính từ sau @ đến hết

c. Code

string findEmailDomain(string address) {

    int i=0;

     do{

         i = address.find('@',i+1);

    }

    while(!isalpha(address[i+1]));

    return address.substr(i+1);

}

**56. HTML End Tag By Start Tag**

a. Đề bài

Cho 1 thẻ Start của HTML, tìm thẻ End

b. Giải pháp

Lấy tên thẻ mở rồi thêm các ký tự “/” và “>” để cho đúng cú pháp

c. Code

string htmlEndTagByStartTag(string startTag) {

    int i = startTag.find\_first\_of(' ');

    string tagname = startTag.substr(0,i);

    tagname.insert(1,"/");

    if(tagname[tagname.length()-1]!='>')

    tagname.append(">");

    return tagname;

}

**57. Is MAC48 Address?**

a. Đề bài

Cho 1 xâu, kiểm tra xem xâu đó có phải địa chỉ MAC-48 không?

b. Giải pháp

-Nếu xâu có độ dài lớn hơn 17 thì không phải là địa chỉ MAC-48

-Duyệt qua từng ký tự trong xâu xem có nằm trong đoạn [A,F], [0,9] và các ký tự ở vị trí 2,5,8,… có phải “-” không?

c. Code

bool isMAC48Address(string inputString) {

    int i=0;

    if(inputString.length()>17) return false;

    do{

        if((inputString[i]<='9'&&inputString[i]>='0')||(inputString[i]<='F'&&inputString[i]>='A')) i++; else return false;

        if((inputString[i]<='9'&&inputString[i]>='0')||(inputString[i]<='F'&&inputString[i]>='A')) i++; else return false;

        if(inputString[i]!='-'&&i<17) return false;

        i++;

    }while(i<17);

    return true;

}

**58. Is Unstable Pair?**

a. Đề bài

Để so sánh 2 chuỗi a và b, ta tìm vị trí i đầu tiên mà a[i] ≠ b[i],nếu a[i] < b[i] thì a<b và ngược lại. Nếu không có vị trí thỏa mãn thì xâu nào ngắn hơn sẽ nhỏ hơn.

Gọi 2 xâu là “unstable” nếu thứ tự so sánh của chúng phụ thuộc vào các trường hợp khác nhau của ký tự (viết hoa hoặc không)

b. Giải pháp

Kiểm tra xem kết quả so sánh 2 xâu đã cho lần đầu.

Chuyển 2 xâu về cùng kiểu chữ thường rồi so sánh 2 xâu.

Nếu kết quả 2 lần là khác nhau thì 2 xâu là unstable.

c. Code

void toLower(string& s) {

    for (int i=0;i<s.size();++i) {

        s[i]=tolower(s[i]);

    }

}

int compare(string& s1, string& s2) {

    for (int i=0;i<s1.size()&&i<s2.size();++i) {

        if (s1[i]>s2[i]) {

            return 1;

        } else if (s1[i]<s2[i]) {

            return -1;

        }

    }

    if (s1.size()>s2.size()) {

        return 1;

    } else if (s1.size()<s2.size()) {

        return -1;

    } else {

        return 0;

    }

}

bool isUnstablePair(string filename1, string filename2) {

    int cmp1 = compare(filename1, filename2);

    toLower(filename1);

    toLower(filename2);

    int cmp2 = compare(filename1, filename2);

    return cmp1!=cmp2;

}

**59. String Construction**

a. Đề bài

Cho 2 chuỗi a và b, cả 2 chỉ bao gồm các chữ cái tiếng Anh viết thường, tính xem có bao nhiêu chuỗi a có thể được xây dựng bằng các chữ cái trong chuỗi b, với mỗi ký tự chỉ được dùng 1 lần?

b. Giải pháp

Duyệt lần lượt các ký tự ở xâu a, nếu tìm thấy ở xâu b thì xóa ký tự đó ở xâu b, đến khi duyệt hết xâu a 1 lần thì tăng biến đếm thêm 1, lặp lại đến khi không tìm thấy ký tự xâu a trong xâu b.

c. Code

#include <stdexcept>

int stringsConstruction(string a, string b) {

    int count =0;

    while(1){

        for(int i=0;i<a.length();i++){

            int pos = b.find(a[i]);

            if(pos>=0) b.erase(b.begin()+pos);

            else {

                return count;

            }

        }

        count ++;

    }

}

**60. Is Substitution Cipher?**

a. Đề bài

Cho 2 xâu a và b có độ dài bằng nhau, mỗi ký tự trong a tương ứng với 1 ký tự trong b, nếu 1 ký tự trong a (hoặc b) tương ứng với nhiều ký tự trong b (hoặc a) thì trả về false, ngược lại trả về true.

b. Giải pháp

Duyệt lần lượt các ký tự trong a đến khi ký tự ở vị trí tương ứng ở a và b khác nhau rồi duyệt lại 1 vòng xem ký tự ở a hoặc b có được thay thế bởi ký tự khác không

c. Code

bool isSubstitutionCipher(string string1, string string2) {

    for(int i=0;i<string1.length();i++){

        if(string1[i]!=string2[i]) {

            for(int j=0;j<string2.length();j++)

                if((string2[j]==string2[i]&&string1[j]!=string1[i])||(string1[j]==string1[i]&&string2[j]!=string2[i])) return false;

        }

}

 return true;

}